Exercises of Chapter 4

Chuang-Chieh Lin*

Department of Computer Science and Information Engineering,
National Chung Cheng University, Ming-Hsiung, Chiayi 621, Taiwan.

Exercise 4.10. *A casino is testing a new class of simple slot machines. Each game, the player puts in \$1, and the slot machine is supposed to return either \$3 to the player with probability* 4/25, *\$100 with probability* 1/200, *or nothing with all remaining probability. Each game is supposed to be independent of other games.*

The casino has been surprised to find in testing that the machines have lost \$10,000 over the first million games. Derive a Chernoff bound for the probability of this event. You may want to use a calculator or program to help you choose appropriate values as you derive your bound.

Solution. Let X_i denote the net loss of the casino for game i, and we denote $X = \sum_{i=1}^{1000000} X_i$, which is the net loss over 1000000 games. By the description of the problem, we know $\mathbf{Pr}[X_i = 2] = 4/25$, $\mathbf{Pr}[X_i = 99] = 1/200$, and $\mathbf{Pr}[X_i = -1] = 1-4/25-1/200 = 167/200$. Since X_i's are mutually independent, we have

$$\begin{aligned}
\mathbf{E}[e^{tX}] &= M_X(t) \\
&= M_{X_1+\ldots+X_{10^6}}(t) \\
&= \prod_{i=1}^{10^6} \mathbf{E}[e^{tX_i}] \\
&= (\mathbf{E}[e^{tX_1}])^{10^6}.
\end{aligned}$$

Then can derive the Chernoff bound for X as follows.
For $t > 0$,

$$\begin{aligned}
\mathbf{Pr}[X \geq 10000] &= \mathbf{Pr}[e^{tX} \geq e^{t\cdot 10000}] \\
&\leq \frac{\mathbf{E}[e^{tX}]}{e^{10^4 t}} \\
&= \frac{(\mathbf{E}[e^{tX_1}])^{10^6}}{e^{10^4 t}} \\
&= \frac{\left(\frac{167}{200}\cdot e^{-t} + \frac{1}{200}\cdot e^{99t} + \frac{32}{200}\cdot e^{2t}\right)^{10^6}}{e^{10^4 t}}
\end{aligned}$$

*Email address: lincc@cs.ccu.edu.tw

Let $f(t) = (\frac{167}{200} \cdot e^{-t} + \frac{1}{200} \cdot e^{99t} + \frac{32}{200} \cdot e^{2t})^{10^6} / e^{10^4 t}$. By using the software Maxima (or, MATLAB), we can obtain that the minimum value of $f(t)$ is larger than 0.000577 and a little bit smaller than 0.000578 (see Fig. 1), and also by Maxima we have $f(0.000577) \approx 0.0001586$. Hence we have $\mathbf{Pr}[X \geq 10000] \leq 0.0001586$. From the point of view of the boss of the casino, we recommend that the slot machines should be checked!

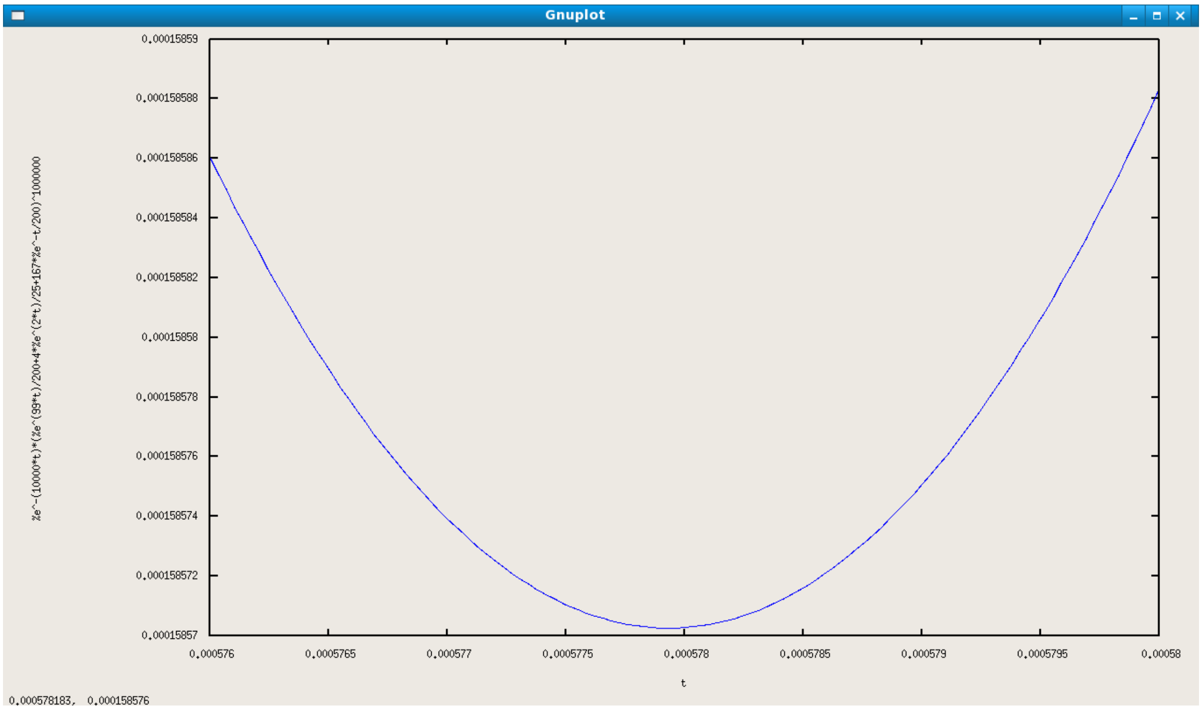

Fig. 1: Gnuplot of $f(t)$ by Maxima.

□

Exercise 4.20. *We prove that the Randomized Quicksort algorithm sorts a set of n numbers in time $O(n \log n)$ with high probability. Consider the following view of Randomized Quicksort. Every point in the algorithm where it decides on a pivot element is called a* node. *Suppose the size of the set to be sorted at a particular node is s. The node is called* good *if the pivot element divides the set into two parts, each of size not exceeding $2s/3$. Otherwise the node is called* bad. *The nodes can be thought of as forming a tree in which the root node has the whole set to be sorted and its children have the two sets formed after the first pivot step and so on.*

(a) *Show that the number of good nodes in any path from the root to a leaf in this tree is not greater than $c \log_2 n$, where c is some positive constant.*

(b) *Show that, with high probability (greater than $1 - 1/n^2$), the number of nodes in a given root to leaf path of the tree is not greater than $c' \log_2 n$, where c' is another constant.*

(c) *Show that, with high probability (greater than $1 - 1/n$), the number of nodes in the longest root to leaf path is not greater than $c' \log_2 n$. (*Hint: *How many nodes are there in the tree?)*

(d) *Use your answers to show that the running time of Quicksort is $O(n \log n)$ with probability at least $1 - 1/n$.*

Solution. (a) Let $D(s)$ denote the depth of tree representing the behavior of Randomized Quicksort algorithm which sorts a set of s numbers. We denote by $N(s)$ the node of the tree which stands for sorting s numbers. Then for the tree node $N(s)$ which has two children $N(a)$ and $N(s-a)$, we have the following recurrence:

$$D(s) = \max\{D(a), D(s-a)\} + 1,$$

where $D(1) = 1$ and $D(s)$ is monotonically nondecreasing with respect to s. Each recursion, say $D(s)$, stands for a node having two children, say $D(a)$ and $D(s-a)$, of the tree. From the description of the problem, we call a node $N(s)$, which has two children $N(a)$ and $N(s-a)$, is good, if $\max\{a, s-a\} \leq 2s/3$, i.e., $\max\{D(a), D(s-a)\} \leq D(2s/3)$. Hence for a good node $N(s)$, we have $D(s) \leq D(2s/3) + 1$. Thus the number of good nodes in any path from the root to a leaf in the tree is at most $\log_{3/2} n = \log_2(2/3) \cdot \log_2 n$. Here $\log_2(2/3) \approx 0.631$ can be chosen to be the desired constant c.

(b) Let $c' = 36$ (i.e., the number of nodes in a given root-to-leaf path of the tree is at least 36) and $\delta = 9/20$. By (a) we know the number of good nodes in any path from the root to a leaf in this tree is not greater than $c \log_2 n$, where $c \approx 0.631$, we obtain that the number of *bad* nodes in the path is at least $35 \log_2 n$. Let X_i be an indicator random variable such that $X_i = 1$ if the ith node in the path is bad, and $X_i = 0$ otherwise. Then what we want to estimate is the probability that $\mathbf{Pr}[X = \sum_{i=1}^{36 \log_2 n} X_i \geq 35.369 \log_2 n]$. Note that a node is good if and only if the chosen pivot is greater than or equal to the $(s/3)$th smallest element, or less than or equal to the $(2s/3)$th smallest element of the current set of s numbers to be sorted. Hence we have $\mathbf{Pr}[X_i = 1] \leq 2/3$, $\mathbf{Pr}[X_i = 0] \geq 1/3$, and $\mathbf{E}[X] \leq (2/3) \cdot 36 \log_2 n = 24 \log_2 n$. Besides, by extending Theorem 4.4 of [2] we have the following corollary (refer to Exercise 4.7 at page 84 of [2]):

> **Corollary 1.** *Let $Y = \sum_{i=1}^{n} Y_i$, where Y_i's are independent 0-1 random variables. Let $\mu = \mathbf{E}[Y]$. Choose any $\mu \leq \mu_H$. Then for any $0 < \delta \leq 1$,*
>
> $$\mathbf{Pr}[Y \geq (1+\delta)\mu_H] \leq e^{-\mu_H \delta^2/3}.$$

Let $\mu_H = 24 \log_2 n$. Therefore we have

$$\begin{aligned}
\mathbf{Pr}[X \geq 35.369 \log_2 n] &\leq \mathbf{Pr}[X \geq 34.8 \log_2 n] \\
&= \mathbf{Pr}[X \geq (1+\delta) \cdot \mu_H] \\
&\leq e^{-\mu_H \delta^2/3} \\
&= e^{-\frac{81}{50} \cdot \frac{\ln n}{\ln 2}} \\
&\leq n^{-2.337} \\
&< n^{-2}.
\end{aligned}$$

Hence we have the desired probability $1 - 1/n^2$.

(c) Note that the number of root-to-leaf paths of the tree is n. Let A_i denote the event that the number of nodes in the ith fixed root-to-leaf path is greater than $c' \log_2 n$ for some constant c' (where c' is chosen to be 36). We have shown that $\mathbf{Pr}[A_i] \leq 1/n^2$. Thus by the union-bound, the probability that the number of nodes in the *longest* root-to-leaf path is greater than $c' \log_2 n$, that is, $\mathbf{Pr}[\bigcup_{i=1}^n A_i]$, is at most $\sum_{i=1}^n \mathbf{Pr}[A_i] = 1/n$. Hence we have the desired probability.

(d) Let $T(n)$ be the running time of the Randomized Quicksort which sorts n numbers, then we have $T(n) = T(a) + T(n-a) + O(n)$, where $O(n)$ comes from comparisons between the pivot with other $(n-1)$ numbers in one recursion. Since the depth of the recursion of the Randomized Quicksort algorithm is at most the number of nodes of the longest root-to-leaf path of the corresponding tree, and it is $O(\log n)$ with probability $1 - 1/n$, by using the recursion-tree method [1] for analyzing the recurrences, we derive that $T(n) = O(n \log n)$ with probability $1 - 1/n$.

□

References

[1] T. H. Cormen, C. E. Leiserson, and R. L. Rivest: *Introduction to Algorithms. 2nd Edition.* The MIT Press, 2001.

[2] M. Mitzenmacher and E. Upfal: *Probability and Computing: Randomized Algorithms and Probabilistic Analysis.* Cambridge University Press, 2005.